LISTING OF AMENDED CLAIMS

The listing of claims below replaces all prior versions and listings of claims.

1.-16. (Canceled)

17. (Currently amended) A computer-implemented method of storing and translating data between ~~from~~ a format of a first data model of a first software component and ~~to~~ a format of a second data model of a second software component, the method comprising:

creating a first schema comprising the first data model of the first software component;

~~integrating the first schema into a data wedge;~~

creating a second schema comprising the second data model of the second software component;

creating a third data model and a data storage in a data wedge by integrating the first schema and the second schema into the data wedge;

receiving a data element in the format of the first data model of the first software component, translating the data element from the format of the first data model of the first software component to the format of the third data model in the data wedge and storing the translated data element in the data storage, by the data wedge; and

retrieving the data element from the data storage and translating the data element into the format of the second data model of the second software component by the data wedge after receiving a request for the data element from the second software component.

~~integrating the second schema into the data wedge;~~

~~populating the data model of the first software component; and~~

~~translating a data element from the format of the data model of the first software component to the format of the data model of the second software component by the data wedge.~~

18. (Currently amended) The method of claim 17, further comprising:

~~triggering an event to notify the second software component of translated data element availability.~~

triggering an event to notify the second software component of the availability of the data element received from the first software component and stored in the data wedge.

19. (Currently amended) The method of claim 17, further comprising:

reading the ~~translated~~ data element translated into the format of the second data model by the second software component.

20. (Currently amended) The method of claim 17, further comprising:

removing an obsolete data element from the first data model of the first software component and causing the data wedge to remove the translated data element from the third data model.

21. (Previously amended) The method of claim 17, further comprising:

creating an instance of the data wedge.

22. (Previously presented) The method of claim 17, wherein the first and second schemas further comprise a name of the data wedge.

23. (Currently amended) The method of claim 17, wherein integrating the first schema into the data wedge includes setting default data elements and data values for the first data model of the first software component.

24. (Currently amended) The method of claim 17, further comprising:

~~modifying a data element in the data model of the first software component.~~

retrieving the data element from the data storage and translating the data element from the format of the third data model to the format of the first data model of the first software component by the data wedge after receiving a request for the data element from the first software component.

25. (Currently amended) A computer system for translating data between ~~from~~ a format of a first data model of a first software component and ~~to~~ a format of a second data model of a second software component, the system comprising:

a processor; and

a memory coupled to said processor, the memory having stored therein data and sequences of instructions which, when executed by said processor, cause said processor to:

create a first schema comprising the first data model of the first software component;

~~integrate the first schema into a data wedge;~~

create a second schema comprising the second data model of the second software component;

create a third data model and a data storage in a data wedge by integrating the first schema and the second schema into the data wedge;

receive a data element in the format of the first data model of the first software component, translate the data element from the format of the first data model of the first software component to a format of the third data model in the data wedge and store the translated data element in the data storage; and

retrieve the data element from the data storage and translate the data element into the format of the second data model of the second software component after receiving a request for the data element from the second software component.

~~integrate the second schema into the data wedge;~~

~~populate the data model of the first software component; and~~

~~translate a data element from the format of the data model of the first software component to the format of the data model of the second software component by the data wedge.~~

26. (Currently amended) The system of claim 25, further comprising instructions which, when executed by said processor, cause said processor to:

~~trigger an event to notify the second software component of translated data element availability.~~

triggering an event to notify the second software component of the availability of the data element received from the first software component and stored in the data storage.

27. (Currently amended) The system of claim 25, further comprising instructions which, when executed by said processor, cause said processor to:

remove an obsolete data element from the first data model of the first software component and remove the translated data element from the third data model.

28. (Previously presented) The system of claim 25, further comprising instructions which, when executed by said processor, cause said processor to:

create an instance of the data wedge.

29. (Currently amended) The system of claim 25, further comprising instructions which, when executed by said processor, cause said processor to:

~~modify a data element in the data model of the first software component.~~

retrieve the data element from the data storage and translate the data element from the format of the third data model to the format of the first data model of the first software component after receiving a request for the data element from the first software component.

30. (Previously presented) The system of claim 25, wherein the first and second schemas further comprise a name of the data wedge.

31. (Currently amended) The system of claim 25, wherein the instructions causing the processor to integrate the first schema into the data wedge include instructions causing the processor to set default data elements and data values for the first data model of the first software component.

32. (Currently amended) A computer system for translating data between ~~from~~ a format of a data model of a first software component and ~~to~~ a format of a data model of a second software component, the system comprising:

a processor; and

a memory coupled to said processor,

wherein said processor is configured to execute a sequence of instructions contained in said memory, the instructions comprising a data wedge including a first schema of the first software component and a second schema of the second software component, the data wedge configured to translate a data element from the format of the data model of the first software component in accordance with the first schema to a data model of the data wedge and when a request is received from the second software component, translate the data element from the format of the data model of the data wedge to the format of the data model of the second software component in accordance with the second schema.

33. (Previously presented) The system of claim 32, wherein said data wedge is further configured to trigger an event to notify the second software component of translated data element availability.